

```

/*
Ant Project. Otago Polytechnic, New Zealand. 2009 3rd Year B.I.T. project for
Jun Cui, Gareth Dorset and Trevor Farquharson
-::~*_.*~:-

Client: Otago Museum

Developers: Jun Cui - 3rd year student Otago Polytechnic
            Gareth Dorset - 3rd year student Otago Polytechnic
            Trevor Farquharson - 3rd year student Otago Polytechnic

Mentors: Patricia Haden - Otago Polytechnic
         Hamish Smith - Otago Polytechnic
         Sam Mann - Otago Polytechnic
*/

/**
 * @author >> Justin Windle
 * Based on the great Mario Klingemann's ColorMatrix class
 *
 * @link >> soulwire.co.uk
 * @version >> V1
 */

/* The code contained below is the original code by Justin Windle, full credit for making the motion tracker
work goes to him
any modifications to the code here will be values of variables required to fine tune for our
purposes.

We had full permission to use and modify this code.*/

package com.soulwire.geom
{
    public class ColourMatrix
    {
        /*
        =====
        | Private Variables | Data Type
        =====
        */

        protected const LUMINANCE_R: Number = 0.212671;
        protected const LUMINANCE_G: Number = 0.715160;
        protected const LUMINANCE_B: Number = 0.072169;
        protected const IDENTITY: Array = [1, 0, 0, 0, 0, 0,
0, 1, 0, 0, 0, 0,
0, 0, 1, 0, 0, 0,
0, 0, 0, 1, 0];

        protected var _matrix: Array;
        protected var _hue: Number;
        protected var _saturation: Number;
        protected var _brightness: Number;
        protected var _contrast: Number;
        protected var _alpha: Number;

        /*
        =====
        | Constructor
        =====
        */

        public function ColourMatrix( matrix:Array = null )
        {
            init(matrix == null ? IDENTITY.concat() : matrix.concat());
        }

        /*
        =====
        | Private Methods
        =====
        */

        protected function init( matrix:Array ):void
        {
            _matrix = matrix;
            setDefaultValues();
        }

        protected function setDefaultValues():void
        {
            _alpha = 100;
            _brightness = 0;
            _contrast = 0;
            _hue = 0;
            _saturation = 0;
        }

        protected function multiply(matrix:Array):void
        {
            var aBuffer:Array = new Array();

```

```

        var n:int = 0;

        for(var i:int = 0; i < 4; i++)
        {
            for(var j:int = 0; j < 5; j++)
            {
                aBuffer[n + j] = matrix[n]
                    * _matrix[j]
                    + matrix[n + 1] * _matrix[j + 5] +
                    matrix[n + 2] * _matrix[j + 10] +
                    matrix[n + 3] * _matrix[j + 15] +
                    (j == 4 ? matrix[n + 4] : 0);

                n += 5;
            }
            _matrix = aBuffer.concat();
        }

    /*
    =====
    | Public Methods
    =====
    */

    public function getMatrix():Array
    {
        return _matrix.concat();
    }

    public function clone():ColourMatrix
    {
        return new ColourMatrix( getMatrix() );
    }

    public function reset():void
    {
        init( IDENTITY.concat() );
    }

    /*
    =====
    | Getters + Setters
    =====
    */

    /* ALPHA */

    public function get alpha():Number { return _alpha; }

    /**
     * Sets the alpha.
     * @param alpha A value between 0 and 100 (0 being 0% alpha, 100 being 100% alpha).
     */

    public function set alpha( a:Number ):void
    {
        var old:Number = _alpha / 100;

        _alpha = a;
        a /= 100;
        a /= old;

        var matrix:Array = [1, 0, 0, 0, 0,
                                0, 1, 0, 0, 0,
                                0, 0, 1, 0, 0,
                                0, 0, 0, a, 0];

        multiply(matrix);
    }

    /* BRIGHTNESS */

    public function get brightness():Number { return _brightness; }

    /**
     * Sets the brightness.
     * @param brightness A value between -100 and 100 (0 being 'no changes').
     */

    public function set brightness( b:Number ):void
    {
        var old:Number = _brightness * (255 / 100);

        _brightness = b;
        b *= (255 / 100);
        b -= old;

        var matrix:Array = [1, 0, 0, 0, b,
                                0, 1, 0, 0, b,
                                0, 0, 1, 0, b,
                                0, 0, 0, 1, 0];

        multiply(matrix);
    }

    /* CONTRAST */

    public function get contrast():Number { return _contrast; }

```

[illegible]